International Journal of Computer Science Engineering
and Information Technology Research (IJCSEITR)
ISSN(P): 2249-6831; ISSN(E): 2249-7943
Vol. 4, Issue 2, Apr 2014, 5-14

INTRUSION DETECTION SYSTEM USING OPTIMAL C4.5 ALGORITHM

SACHIN PRAKASH GAVHANE[1] & VIJAY MARUTI SHELAKE[2]

[1]Department of Information Technology, Atharva College of Engineering, Malad, Mumbai, Maharashtra, India

[2]Department of Information Technology, YTCEM, Karjat, Maharashtra, India

ABSTRACT

With hacker attacks against well-known businesses and organizations on the rise, network security has made headlines .of course, there are many attacks that do not make headlines and are not reported due to a loss of credibility or embarrassment. Then there are the attacks that are not even detected. The Defense Information Services Agency (DISA) states that up to 98% of attacks go unnoticed. These revelations have caused many businesses to rethink or to start thinking about the security of their own networks. Maximum Processing computation and more time consuming task has always been a limit in processing huge network intrusion data. This problem can be minimized through feature selection to condense the size of the network data involved. Decision tree models allow one to detect anomalies in large databases. In proposed system, we first preprocess dataset (KDD 99 cup). Then we study and analysis of decision tree algorithms (C4.5 and its extension) of data mining for the task of detecting intrusions and compare their relative performances. Based on this study, it can be concluded that even extended C4.5 is complex but decision tree obtained is the most suitable with high true positive (correct detection of attacks) and low false positive (Incorrect detection) with high accuracy.

KEYWORDS: IDS, Security, Attacks, Abnormal Instance

INTRODUCTION

Intrusion detection systems (IDSs) have become a widely used measure for security systems. The main problem for those systems results is the irrelevant alerts on those results. We will propose a data mining based method for classification to distinguish serious alerts and irrelevant one with a performance of 99.9% which is better in comparison with the other recent data mining methods that have reached the performance of 97%. A ranked alerts list also created according to alerts importance to minimize human interventions. The development of IDS (Intrusion Detection System), and the current and gives a brief introduction to DM (Data Mining) technology. Presented a framework of IDS based on data mining for resolving the current problems IDS is facing. The system that performs anomaly detection can detect intrusions known and unknown, reduce omissions and misstatements, improve accuracy and speed of intrusion detection and has good adaptive capacity and scalability.

Security of network systems is becoming increasingly important as more and more sensitive information is being stored and manipulated online. Intrusion Detection Systems. (IDSs) have thus become a critical technology to help protect these systems. Most IDSs are based on hand-crafted signatures that are developed by manual encoding of expert knowledge. These systems match activity on the system being monitored to known signatures of attacks. The major problem with this approach is that these IDSs fail to generalize to detect new attacks or attacks without known signatures. Recently, there has been an increased interest in data mining based approaches to building detection models for IDSs. These models generalize from both known attacks and normal behavior in order to detect unknown attacks They can also

be generated in a quicker and more automated method than manually encoded models that require difficult analysis of audit data by domain experts. Several effective data mining techniques for detecting intrusions have been developed many of which perform close to or better than systems engineered by domain experts.

However, successful data mining techniques are themselves not enough to create deployable IDSs. Despite the promise of better detection performance and generalization ability of data mining-based IDSs, there are some inherent difficulties in the implementation and deployment of these systems. We can group these difficulties into three general categories: accuracy (i.e., detection performance), efficiency, and usability. Typically, data mining-based IDSs (especially anomaly detection systems) have higher false positive rates than traditional hand-crafted signature based methods, making them unusable in real environments. Also these systems tend to be inefficient (i.e., computationally expensive) during both training and evaluation. This prevents them from being able to process audit data and detect intrusions in real time. Finally, these systems require large amounts of training data and are significantly more complex than traditional systems. In order to be able to deploy real time data mining-based IDSs, these issues must be addressed.

Types of IDS

Active and Passive IDS

An active Intrusion Detection Systems (IDS) is also known as Intrusion Detection and Prevention System (IDPS). Intrusion Detection and Prevention System (IDPS) is configured to automatically block suspected attacks without any intervention required by an operator. Intrusion Detection and Prevention System (IDPS) has the advantage of providing real-time corrective action in response to an attack. A passive IDS is a system that's configured to only monitor and analyze network traffic activity and alert an operator to potential vulnerabilities and attacks. A passive IDS is not capable of performing any protective or corrective functions on its own. *Network Intrusion detection systems (NIDS) and Host Intrusion detection systems (HIDS*

Network Intrusion Detection Systems (NIDS) usually consists of a network appliance (or sensor) with a Network Interface Card (NIC) operating in promiscuous mode and a separate management interface. The IDS is placed along a network segment or boundary and monitors all traffic on that segment. A Host Intrusion Detection Systems (HIDS) and software application (agents) installed on workstations which are to be monitored. The agents monitor the operating system and write data to log files and/or trigger alarms. A host Intrusion detection systems (HIDS) can only monitors the individual workstations on which the agents are installed and it cannot monitor the entire network. Host based IDS systems are used to monitor any intrusion attempts on critical servers.

The drawbacks of Host Intrusion Detection Systems (HIDS) are

- Difficult to analyze the intrusion attempts on multiple computers.
- Host Intrusion Detection Systems (HIDS) can be very difficult to maintain in large networks with different operating systems and configurations
- Host Intrusion Detection Systems (HIDS) can be disabled by attackers after the system is compromised.

Knowledge-based (Signature-based) IDS and behavior-based (Anomaly-based) IDS

A knowledge-based (Signature-based) Intrusion Detection Systems (IDS) references a database of previous attack signatures and known system vulnerabilities. The meaning of word signature, when we talk about Intrusion Detection

Systems (IDS) is recorded evidence of an intrusion or attack. Each intrusion leaves a footprint behind (e.g., nature of data packets, failed attempt to run an application, failed logins, file and folder access etc.). These footprints are called signatures and can be used to identify and prevent the same attacks in the future. Based on these signatures Knowledge-based (Signature-based) IDS identify intrusion attempts. The disadvantages of Signature-based Intrusion Detection Systems (IDS) are signature database must be continually updated and maintained and Signature-based Intrusion Detection Systems (IDS) may fail to identify unique attacks. A Behavior-based (Anomaly-based) Intrusion Detection Systems (IDS) references a baseline or learned pattern of normal system activity to identify active intrusion attempts. Deviations from this baseline or pattern cause an alarm to be triggered. Higher false alarms are often related with Behavior-based Intrusion Detection Systems (IDS).

Data Mining

Data Mining (DM), also called Knowledge-Discovery and Data Mining, is one of the hot topic in the field of knowledge extraction from database. Data mining is used to automatically learn patterns from large quantities of data. Mining can efficiently discover useful and interesting rules from large collection of data. It is a fairly recent topic in computer science but utilizes many older computational techniques from statistics, information retrieval, machine learning and pattern recognition. Data mining is disciplines works to finds the major relations between collections of data and enables to discover a new and anomalies behavior. Data mining based intrusion detection techniques generally fall into one of two categories; misuse detection and anomaly detection. In misuse detection, each instance in a data set is labeled as ‘normal’ or ‘intrusion’ and a learning algorithm is trained over the labeled data. These techniques are able to automatically retrain intrusion detection models on different input data that include new types of attacks, as long as they have been labeled appropriately.

The successful data mining techniques are themselves not enough to create deployable IDSs. Despite the promise of better detection performance and generalization ability of data mining-based IDSs, there are some inherent difficulties in the implementation and deployment of these systems. In this paper, we discuss several problems inherent in developing and deploying a real-time data mining-based IDS and present an overview of our research, which addresses these problems. These problems are independent of the actual learning algorithms or models used by IDS and must be overcome in order to implement data mining methods in a deployable system.

RELATED WORK

There are three main components of IDS: data collection, detection, and response. The data collection component is responsible for collection and pre-processing data tasks: transferring data to a common format, data storage, and sending data to the detection module. IDS can use different data sources which are the inputs to the system: system logs, network packets, etc. If an IDS monitors activities on a host and detects violations on the host, it is called host-based IDS (HIDS).

An IDS that monitors network packets and detects network attacks is called network-based IDS (NIDS). NIDSs generally listen in promiscuous mode to the packets in a segment of the network, allowing them to detect distributed attacks. There are also intrusion detection systems that use both host-based IDS and network-based IDS. For example, a system can use NIDS and also HIDS for important hosts in the networks such as servers, databases, and the like. Since NIDS cannot monitor encrypted packets, a hybrid approach, network node IDS (NNIDS) is introduced where each host in the network has NNIDS to monitor network packets directed to the host.

A data mining-based IDS is significantly more complex than a traditional system. The main cause for this is that data mining systems require large sets of data from which to train. The hope to reduce the complexity of data mining systems has led to many active research areas. First, once new data has been analyzed, models need to be updated. It is impractical to update models by retraining over all available data, as retraining can take in particular time, and updated models are required immediately to ensure the protection of extended modified systems. Some mechanism is needed to adapt a model to incorporate new information. Third, many data mining-based IDSs are difficult to deploy because they need a large set of clean (i.e., not noisy) labeled training data. Typically the attacks within the data must either be manually labeled for training signature detection models, or removed for training anomaly detection models. Manually cleaning training data is expensive, especially in the context of large networks. In order to reduce the cost of deploying a system, it must be able to minimize the amount of clean data that is required by the data mining process.

GENERAL METHODOLOGY

Figure 1 shows general approach towards the intrusion detection system, as shown in figure data traffic is given to preprocessing phase which then cleans and classifies the traffic and pass it to data mining algorithm phase where algorithms are applied and results are plotted. According to the plotted results conclusion is made.

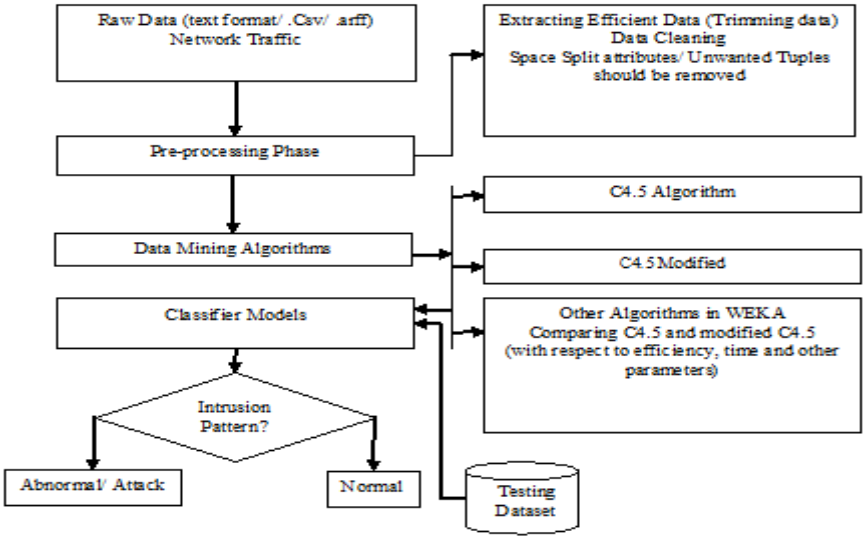

Figure 1: Flow of Proposed System

Table 1: List of Attributes in KDD 99 Cup Dataset

Duration	0.0	is_guest_login	0
protocol type	Tcp	is_host_login	0
Service	http	srv_count	5.0
Flag	SF	serror_rate	0.2
src_bytes	232.0	srv_serror_rate	0.2
dst_bytes	8153.0	rerror_rate	0.0
Land	0	srv_rerror_rate	0.0
wrong fragment	0.0	Same srv rerror rate	1.0
Urgent	0.0	diff_srv_rate	0.0
Hot	0.0	srv_diff_host_rate	0.0
num_failed_logins	0.0	count	5.0
logged_in	1	dst_host_count	30

Table 1: Contd.,

num_compromised	0.0	dst_host_srv_count	255
root_shell	0.0	dst_host_same_srv_rate	1.0
su_attempted	0.0	dst_host_diff_srv_rate	0.0
num_root	0.0	dst_host_same_src_port_rate	0.03
num_file_creations	0.0	dst_host_srv_diff_host_rate	0.04
num_shells	0	dst_host_serror_rate	0.03
num_access_files	0.0	dst_host_srv_serror_rate	0.01
num_outbound_cmds	0.0	dst_host_rerror_rate	0.0
		dst_host_srv_rerror_rate	0.01
		Class← for training (labeled attribute) normal	

DATASET DESCRIPTION

The data set taken here is called KDD CUP 99 dataset which is a record of seven day military traffic prepared and managed by MIT Linchon laboratory. The data set is derived from the 1998 DARPA Intrusion Detection Evaluation program. The dataset consists of 41 attributes including the output attribute and 62234 selected entries. The attribute name and description of each attribute can be given as follows in table 1. The dataset consists of 41 attributes including the output attribute. The dataset consists of numerous entries and thus data mining is required for managing it. This data can be in any format e.g. .txt, .csv, .af

Pre-Processing Phase

The pre-processing phase consists of various operations like data-trimming, data-cleaning, etc.

- **Trimming Phase:** This phase deals with extracting appropriate amount of entries for a system as actual network traffic for a system can be too large at times. The number of entries of normal, each type of abnormal data must be equal. As this dataset will be used for training the system.
- **Cleaning Phase:** The cleaning phase is an important phase as it removes redundant entries by using any suitable data mining tool or a simple java program. Also in this phase the missing values are replaced by appropriate null values.

Data Mining Algorithms

In this phase the data mining algorithms are used for training the system. As here a large dataset is involved, various mining techniques like classification, normalization, etc are used. The proposed system uses two data mining algorithms- Standard C4.5, Modified C4.5 algorithm.

Standard C4.5 Algorithm

The C4.5 algorithm is based on the ID3 algorithm that tries to find small (or simple) Decision Tree's. This process uses the "Entropy", i.e. a measure of the disorder of the data. The Entropy of ~y is calculated by equation (1) as follows:

$$\text{Entropy}(\sim y) = -\sum_{j=1}^{N} |yj| / |\sim y| \log |yj| / \quad (1)$$

Iterating over all possible values of ~y. The conditional Entropy is

Entropy (j|~y) = |yj | / |~y| log (|yj | / |~y|)

And finally, we define Gain by

Gain (~y, j) = Entropy (~y − Entropy (j|~y))

The aim is to maximize the Gain, dividing by overall entropy due to split argument ~y by value j. The algorithm generates a decision tree based on the entropy calculation. The attribute with the highest entropy is placed at the root node. And subsequent values of the selected attributes are used for branching. The number of branches originating from a selected attribute is equal to the number of different values it has. Thus the algorithm will generate an n-ary tree.

Modified C4.5

This algorithm also takes the same equations to calculate the entropy and gain of attribute i.e.

Entropy (~y) = -∑ |yj| / |~y| log| yj / ~y |

Gain (~y, j) = Entropy (~y − Entropy (j|~y))

But the difference here lies in building the decision tree. This algorithm creates a binary decision tree. If we split an attribute in only two subclasses, the resulting entropy will be higher than when it is split into more subclasses. This algorithm is developed for a particular problem in which it was not desirable to split into many subclasses using a single attribute. Therefore the root node in both the cases will be different. As this algorithm now results into a decision tree with a larger depth than previous one, the amount of filtration of the network traffic dataset traversing through it will be more. The classifier model will be different for the standard as well as the modified algorithm because both of them have a different decision tree. On applying the testing dataset to the model one can identify normal, intruder user and thus obtain the number of false positive, false negative, time taken to identify the user.

EXPERIMENTAL RESULTS

Confusion Matrix

Some part of the dataset is used for training and rest of the part is used for testing. The total number of instances provided as input to the decision tree will be predicted as correct or incorrect and the performance will be measured based on number of entries correctly detected. The Advantage of using this matrix is that it not only tells us how many got misclassified but also what misclassifications occurred. The table 2 shown below is the performance classifier for the system also called the confusion matrix.

Table 2: Confusion Matrix

Confusion Matrix		Predicted Class	
		Normal	**Intrusion/Attack**
Actual Class	Normal	True Negative	False Positive
	Intrusion / Attack	False Negative	Correctly Detected

- **True Positive:** The number of attacks correctly identified.
- **True Negative:** The number of normal instances correctly identified.
- **False Positive:** The number of normal records incorrectly identified.

- **False Negative:** is the number of attacks incorrectly identified.

Detection Rate

Same dataset is given to both algorithms and the numbers of detected instances are compared, which is shown in Table 3.

Table 3: Detection Rate of Classification Algorithms

Sr. No.	Algorithm	Normal	Smurf	Neptune
1	Actual entries	205	174	198
2	Standard C4.5	206	170	201
3	Modified C4.5	204	174	199

Comparison Parameters

The same dataset is given as input to both algorithms and the comparison is made between them based on the parameters like time taken, false positives, false negatives detected, which is shown in Table 4.

Table 4: Comparing Standard C4.5 and Modified C4.5

Sr. No.	Algorithm	Time Taken(Sec)	False Positive (%)	False Negative (%)
1	Standard C4.5	2	3.466	3.639
2	Modified C4.5	473	0.5199	0.3466

Table 5 shows comparison of Standard C4.5, Modified C4.5 and Naïve Bayes (in Weka) with respect to different parameters.

Table 5: Comparison of Standard C4.5 / Modified C4.5/ Naïve Bayes (In Weka)

	False Positive (%)	False Negative (%)	Time to Build Model (in Sec)	Time for Testing (in Sec)	Incorrectly Classifed Instances (%)	Correctly Classified Instances (%)
Std.C4.5	3.4	3.7	4	0	8	92
Naïve Bayes –Weka	2.97	29.91	1.03	1.69	24	76
Mod. C4.5	0.51	0.34	80	0	1	99

Table 6 shows comparison of J48 (in Weka) / Naïve Bayes in Weka.

Table 6: Comparison of Algorithms in Weka Tool

	False Positive (%)	False Negative (%)	Time to Build Model (in Sec)	Time for Testing (%)	Incorrectly Classified Instances (%)	Correctly Classified Instances (%)
Naïve Bayes	27.47	3.9	1.03	1.17	24	76
J48	1.43	28.21	37.99	0.44	18	82

Graph Analysis

The dataset of KDD cup'99 contains 5,000,000 packets. Handling of such huge data requires some data mining technique. There are various decision tree algorithms like J48, Random Forest, LAD Tree, etc. In this paper, we compare decision tree algorithms. Obtained results are self explanatory and are shown in Figure 2, 3, 4 and 5 respectively.

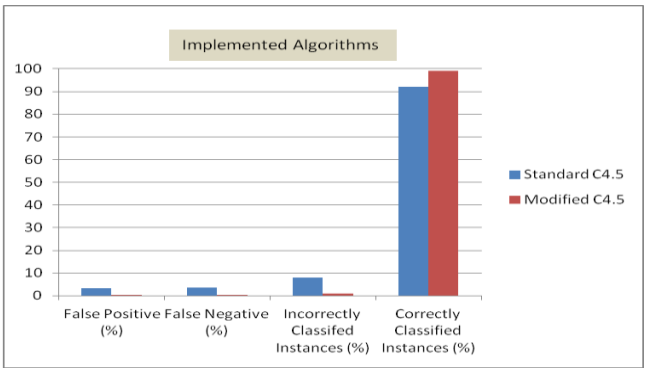

Figure 2: Comparisons of Standard C4.5 and Modified C4.5

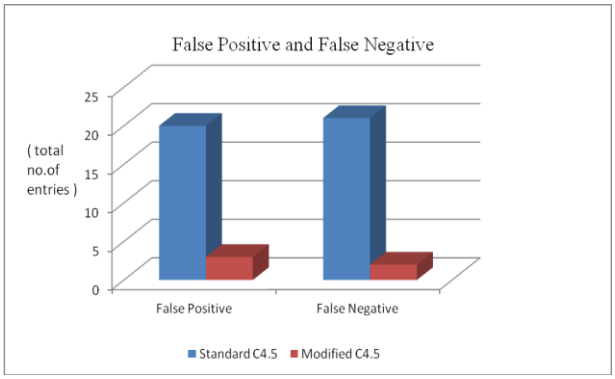

Figure 3: False Positive / False Negative by Implemented Algo

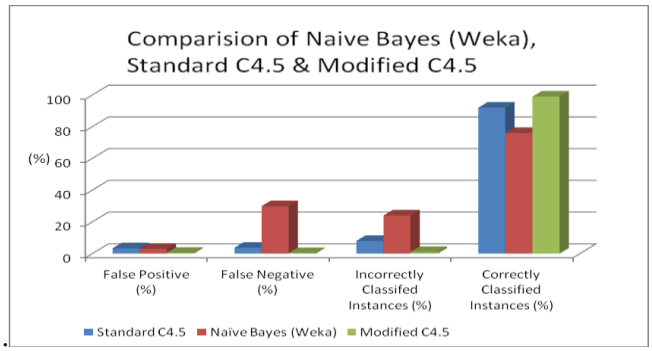

Figure 4: Comparison of Standard C4.5 / Modified C4.5 /

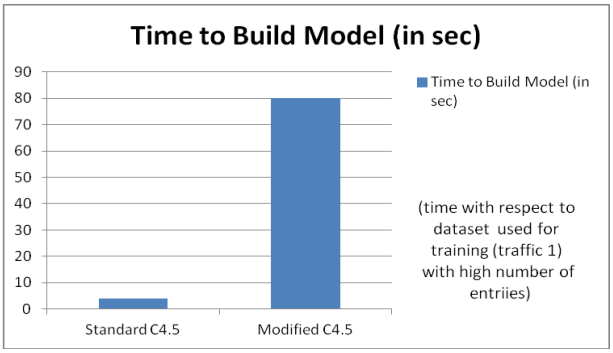

Figure 5: Time Taken to Build Decision Tress Naïve Bayes (In Weka)

CONCLUSIONS

We concluded that modified C4.5 algorithm got promising results compared to standard C4.5. The presented algorithm makes the constructed decision tree more clear and understandable. Also we can conclude that some pre processing can enhance the training time performance. Training time is increasing as number of packets goes on increasing. The system produced satisfies the requirements outlined at an early stage in the project. We provided a graphical user interface with the possibility of user interaction where users will have access to the different functionalities and can follow up on the process of training/testing. The system is used to analyze the training time, accuracy, false positive and false negative rates on training sets of variable sizes

ACKNOWLEDGEMENTS

We would like to thank Prof. Varunakshi Bhojane, Head of Department, PIIT, New Panvel for facilitating all the necessary inputs, study material and resources and guiding me with their rich experience. We would especially like to thank our parents and friends for their unconditional support.

REFERENCES

1. N.s.chandolikar & v.d.nandavadekar, International Journal of Computer Science and Engineering (IJCSE), Vol.1, Issue 1 Aug 2012 81-88 “comparative analysis of two algorithms for intrusion attack classification using kdd cup dataset”

2. Dai Hong Li Haibo,2009 15th IEEE Pacific Rim International Symposium on Dependable Computing, A Lightweight Network Intrusion Detection Model Based on Feature Selection
3. Bhavani Thuraisingham, Latifur Khan, Mohammad M. Masud, Kevin W. Hamlen The University of Texas at Dallas 2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing," Data Mining for Security Applications"
4. James Cannady Jay Harrell "A Comparative Analysis of Current Intrusion Detection Technologies"
5. Mrudula Gudadhe, Prakash Prasad, Kapil Wankhade, Lecturer, "a new data mining based network intrusion detection model"iccct'10
6. Radhika Goel, Anjali Sardana, and Ramesh C. Joshi "Parallel Misuse and Anomaly Detection Model" International Journal of Network Security, Vol.14, No.4, PP.211-222, July 2012
7. Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. GhorbaniA Detailed Analysis of the KDD CUP 99 Data Set
8. Thales Sehn Korting," C4.5 algorithm and Multivariate Decision Trees"
9. P Amudha, H Abdul Rauf, "Performance Analysis of Data Mining Approaches in Intrusion Detection"
10. Knowl Inf Syst (2008) 14:1–37 DOI 10.1007/s10115-007-0114-2 SURVEY PAPER, Top 10 algorithms in data mining XindongWu · Vipin Kumar · J. Ross Quinlan · Joydeep Ghosh · Qiang Yang ·Hiroshi Motoda Geoffrey J. McLachlan · Angus Ng · Bing Liu · Philip S. Yu ·Zhi-Hua Zhou · Michael Steinbach · David J. Hand Dan Steinberg
11. Mrutyunjaya Panda1 and Manas Ranjan Patra 2 , IJCSNS International Journal of Computer Science and Network Security, VOL.7 No.12, December 2007" Network Intrusion Detection Using Naïve Bayes"
12. Shaik Akbar, Dr. K. Nageswara Rao, Dr. J. A. Chandulal, International Journal of Computer Applications (0975 –8887) Intrusion Detection System Methodologies Based on Data Analysis
13. Nathan Einwechter, "An Introduction To Distributed Intrusion Detection Systems"
14. Payam Emami Khoonsari and AhmadReza Motie, "A Comparison of Efficiency and Robustness of ID3 and C4.5,Algorithms Using Dynamic Test and Training Data Sets", International Journal of Machine Learning and Computing, Vol. 2, No. 5, October 2012
15. http://kdd.ics.uci.edu/databases/kddcup99/task.html
16. http://nsl.cs.unb.ca/NSL-KDD/
17. http://www.coli.unisaarland.de/~crocker/Teaching/Connectionist/lecture9_4up.pdf